Estrutura de Dados 2

Exercício

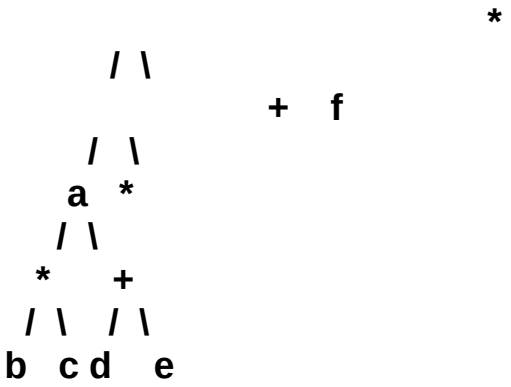
Baseado no material disponibilizado em

https://www.ime.usp.br/~pf/mac0122-2002/aulas/trees.html

Árvore de expressão aritmética

Vamos considerar aqui mais um exemplo de construção de árvore binária. Desta vez, a árvore será uma representação de uma expressão aritmética.

Suponha que temos uma expressão aritmética cujos operadores são todos binários. Mais concretamente, suponha que os operadores são soma (+) e multiplicação (*). Suponha também, para simplificar, que os operandos são nomes de variáveis, cada um consistindo de uma única letra. **Uma expressão aritmética pode ser muito bem representada por uma árvore binária: as folhas da árvore são operandos e os nós internos são operadores.**



Lembrem-se: se a árvore for lida em ordem esquerda-**raiz**-direita, teremos a expressão em notação infixa. Se for lida em ordem esquerda-direita-**raiz**, teremos a expressão em notação posfixa. Se for lida em ordem **raiz**-esquerda-direita, teremos a expressão em notação prefixa.

infixa	`(a + (b*c) * (d+e))* f`
posfixa	`abc * de +*+ f *`
prefixa	`*+ a ** bc + def`

Se a expressão consiste em uma única letra, a árvore terá um único nó; se a expressão for algo como `*ab`, a árvore terá uma raiz e duas folhas.

Exemplo de código:

Suponha que a expressão **prefixa** está armazenada em um vetor global de caracteres `a[i], sendo i` uma variável global.

```
typedef struct Tnode *link;

struct Tnode {
  char token;
   link l, r;
 } ;

char *a;
int i;

       // A função parse atua sobre a expressão prefixa a[i..].
       // Os operadores são '+' e '*', cada variável tem
       // um só caracter, e não há espaços entre os caracteres.
       // A função transforma a expressão em uma árvore binária
       // e devolve a raiz da árvore.
```

```
link parse() {
    char t;
   link x;

  t = a[i++];
    x = malloc(sizeof *x);
    x->token = t;
  if (t == '+' || t == '*') {
         x->l = parse();
         x->r = parse();
  }
   else {
    x->l = NULL;
    x->r = NULL;
  }
    return x;
}
```

1. "Inspirando-se" no exemplo acima, escreva uma função que receba uma expressão aritmética e construa a correspondente árvore. Suponha que a expressão só envolve os operadores `'+','-','*',` e `'/'` e operandos que consistem em uma só letra.

2. Escreva uma função que calcule o valor da expressão aritmética representada por uma árvore sendo dados os valores das variáveis. Suponha que os valores das variáveis são dados em um vetor do tipo **`int`** indexado por letras (ou seja, uma tabela de símbolos-valor).

Observações:

- Escolha qual ordem de percurso é mais adequada
- Determine como será o tipo *link*
- Note que a árvore é heterogênea: há valores numéricos (operandos) e caracteres (operadores). Trate-os adequadamente.